VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

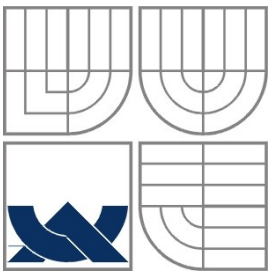
ŠIFROVÁNÍ TELEFONNÍCH HOVORŮ

DIPLOMOVÁ PRÁCE
MASTER´S THESIS

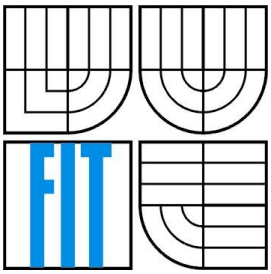
AUTOR PRÁCE
AUTHOR

BC. JAKUB VÁVRA

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

ŠIFROVÁNÍ TELEFONNÍCH HOVORŮ

ENCRYPTING LANDLINES PHONE COMMUNICATION

DIPLOMOVÁ PRÁCE
MASTER´S THESIS

AUTOR PRÁCE
AUTHOR

BC. JAKUB VÁVRA

VEDOUCÍ PRÁCE
SUPERVISOR

DOC. ING. DANIEL CVRČEK, PH.D.

BRNO 2008

Abstrakt

Diplomová práce se zabývá navržením a vhodnou implementací šifrování telefonního hovoru pomocí desky FITkitu. Cílem diplomové práce je zvolit vhodné kompresní a šifrovací algoritmy a implementovat je, či případně přizpůsobit pro desku FITkitu.

Klíčová slova

Pevná linka, šifrování hovoru, kodeky, vokodéry, xor, AES, Skipjack, IDEA, TEA, Blowfish, PCM, ADPCM, GSM, A-law, MU-law, CELP, FITkit.

Abstract

This master's thesis is about making draft and implementing land-line phone call encryption using FITkit. The ultimate goal is to find suitable compression and encryption methods, implement or adapt them for FITkit board and create functional solution.

Keywords

Land-line, encrypting calls, codecs, vocoders, xor, AES, Skipjack, IDEA, TEA, Blowfish, PCM, ADPCM, GSM, A-law, MU-law, CELP, FITkit.

Citace

VÁVRA, Jakub, Bc.: *Šifrování telefonních hovorů*, diplomová práce, Brno, FIT VUT v Brně, 2008

Šifrování telefonních hovorů

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením:

doc. Ing. Daniela Cvrčka, Ph.D.

Další informace mi poskytli členové FITkit týmu.

Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal. Informace o hardwaru a softwaru FITkitu vycházejí částečně z mé bakalářské práce [BPO] a nebudou označovány jako citace.

……

Jakub Vávra

Datum

Poděkování

Na tomto místě bych chtěl poděkovat všem, kteří mne při této práci podporovali ať už morálně nebo nebo moudrými radami a zajímavými postřehy. Děkuji vedoucímu práce doc. Danielu Cvrčkovi bez jehož vedení by tato práce nemohla vzniknout. Zvláštní poděkování patří oponentu Mgr. Marku Kumpoštovi za cenné konzultace ohledně formální stránky diplomové práce. Poděkování patří i mé přítelkyni Mgr. Kateřině Němcové za korektury zkvalitňující gramatickou stránku práce.

Obsah

Seznam ilustrací....3
Seznam tabulek....3
1 Úvod....4
1.1 Specifikace výsledného zařízení....4
1.2 Komprese zvuku....5
1.3 Šifrování....5
1.4 FITkit....6
1.5 Problematika telefonních linek....6
1.6 Modem....6
2 Platforma FITkit....7
2.1 Periferie....8
2.2 Mikrokontrolér....8
2.2.1 Převodník ADC12....9
2.2.2 Převodník DAC12....10
2.3 FPGA....11
2.4 Rozhraní....11
3 Analýza problému....12
3.1 Mapování systému....12
3.2 Komprese zvuku....14
3.2.1 Vzorkování....14
3.2.2 Zvukové kodeky a formáty....14
3.3 Šifrovací algoritmy....17
3.4 Modem....18
3.5 Ovládání aplikace....18
3.6 Komunikační protokol....18
4 Implementace....19
4.1 Mapování systému....19
4.2 Vzorkování a komprese....19
4.3 Šifrování....19
4.4 Spojovací systém FITkitu – SPI....20
4.5 Periferie na FPGA....20
4.5.1 Klávesnice 4x4....20
4.5.2 LCD Display....21

4.5.3 Řadič RS232 ... 22
4.6 Ovládání aplikace ... 22
4.6.1 Klávesnice 4x4 ... 22
4.6.2 Příkazy terminálu ... 23
4.7 Implementační problémy ... 25
4.7.1 Velikost paměti ... 25
4.7.2 Ladicí výpisy ... 25
5 Testování systému ... 26
5.1 Testy komprese zvuku ... 26
5.2 Testy šifrování ... 26
5.3 Testovací zapojení ... 26
5.3.1 První fáze – testování komprese a šifrování ... 26
5.3.2 Druhá fáze – testování komunikace po telefonní lince ... 27
5.4 Kvalita zvuku ... 27
5.5 Úzká hrdla ... 27
6 Hardwarová implementace ... 28
6.1 Změny v hardwaru ... 28
6.2 Změny v softwaru ... 28
7 Závěr ... 30
Literatura ... 31
Rejstřík zkratek ... 34
Přílohy ... 35
Příloha A: Příkazy LCD displeje ... 35
Příloha B: BSD Licence ... 36
Příloha C: Šifrovací zařízení ... 37
Enigma 100 ... 37
TSD 3600 ... 37
ELE-900 ... 37
SV-100 Safe Voice Scrambler ... 37
Phone Line Scrambler Call Encrypter Kit TCS-E1 ... 37
KV1200 Landline Scrambler ... 37
Příloha D: Struktura adresářů ... 38
Příloha E: Nahrávání programu ... 39
Nástroje ... 39
Kompilace konfiguračního souboru pro FPGA ... 39

Kompilace programu pro MCU 39
Nahrání konfigurace do FPGA 39

Seznam ilustrací

Kresba 1: Užitý modem DeskPorte Home 6
Kresba 2: Deska FITkitu, rozmístění komponent a rozhraní [FKM] 7
Kresba 3: Blokové schéma MSP430F1611 [TI1] 8
Kresba 4: Schéma vnitřního uspořádání FPGA [X2] 11
Kresba 5: Hrubé schéma navrhovaného systému 12
Kresba 6: Rozdělení funkčních bloků systému - varianta 1 13
Kresba 7: Rozdělení funkčních bloků systému - varianta 2 13
Kresba 8: Rozdělení funkčních bloků systému - varianta 3 13
Kresba 9: Diagram klávesnice 4x4 [MCD] 21
Kresba 10: Rozhraní LCD displeje [DIC] 21
Kresba 11: Terminál FITkitu po startu 24

Seznam tabulek

Tabulka 1: Organizace paměti mikrokontroléru msp430f1611 9
Tabulka 2: Přehled kodeků a formátů 16
Tabulka 3: Přiřazení pinů klávesnice 21
Tabulka 4: Přehled příkazů LCD [DIC] 35

1 Úvod

V dnešní době stále nabývá důležitost soukromé a důvěrné komunikace. Tento trend se projevuje například ve formě šifrování v mobilních telefonech i nárůstem využití dalších kryptografických metod. I když se značně rozšířila technologie GSM (Global System for Mobile communications) a VoIP (Voice over Internet Protocol), stále se hojně využívá klasická pevná linka.

Na trhu v současné době existuje několik různých zařízení pro šifrování hovoru, ale buď jsou nedostatečně bezpečná (Enigma 100, ASC-2), nebo jsou sledovatelná vládou USA (TSD 3600). U mnoha zařízení není zmíněna šifrovací metoda, takže se jedná o bezpečnost skrze utajení algoritmu, což je dávno překonaný princip. Podrobně zde nebudu jednotlivá zařízení rozebírat, pouze jsem mezi přílohy připojil odkazy na zdroje informací o těchto výrobcích (Příloha C: Šifrovací zařízení) [AMZ].

Diplomová práce se zabývá šifrováním telefonních hovorů. K vzorkování, kompresi a šifrování zvuku budeme užívat desku FITkit.

Data budeme přenášet pomocí modemu skrze analogovou telefonní linku. Předpokládáme, že na obou koncích linky bude naše zařízení.

1.1 Specifikace výsledného zařízení

Výsledné kryptografické zařízení má být založeno na mikrokontroléru osazeném na FITkitu. Produkt bude zapojován mezi telefon a linku a bude provádět kompresi a šifrování zvuku. Zařízení bude umožňovat volbu klíče pro šifrování, stav zařízení bude indikován pomocí LCD displeje a několika LED diod.

Produkt bude osazen následujícími rozhraními:

- RJ 11 – zásuvka pro připojení telefonní linky.
- RJ 11 – zásuvka pro připojení telefonu.
- USB – zásuvka USB B nebo Mini USB pro případnou správu klíčů zařízení z PC.
- Zásuvka pro napájecí adaptér.

Ovládání:

- LCD displej pro zobrazení aktuálního klíče, menu umožňující výběr přednastavených klíčů.
- LED indikující napájení.
- LED indikující zapnutí přístroje a jeho připravenost.
- LED indikující zapnuté šifrování hovoru.

Klávesnice 4x4 pro zadávání klíčů.

Přepínač pro zapnutí a vypnutí zařízení.

Přepínač spínající šifrovaný mód.

1.2 Komprese zvuku

Zvuk je každé podélné (v pevných látkách případně také příčné) mechanické vlnění v látkovém prostředí, které je schopno vyvolat v lidském uchu sluchový vjem. Frekvence tohoto vlnění leží v rozsahu přibližně 20 Hz až 20 kHz; za jeho hranicemi člověk zvuk sluchem nevnímá [WP1].

Jak je vidět, tak zvuk je analogová v čase spojitá veličina, bohužel pro účely digitálního zpracování je třeba spojitou veličinu nahradit nespojitou. Proces při němž zachycujeme úroveň veličiny pouze v určitých okamžicích nazýváme vzorkování. Jelikož nejsme schopni uložit přesnou hodnotu analogové veličiny, vezmeme vždy hodnotu nejbližší hladiny. Počet hladin – úrovní je dán počtem bitů, na něž veličinu zapisujeme a také schopnostmi AD převodníku (Analog Digital). Tento způsob záznamu zvuku se nazývá pulzně kódová modulace (PCM). Vhodná frekvence pro snímání signálu, aby byl dostatečně rekonstruovatelný, se dle Nyquistova[I] teorému rovná dvojnásobku frekvence vzorkovaného signálu. Takže pro lidským uchem slyšitelný zvuk je třeba vzorkovat alespoň na 40kHz. V praxi se používají různé frekvence na základě požadované kvality zvuku. Další metody komprese zvuku budou uvedeny dále spolu s podrobnostmi o vzorkování.

1.3 Šifrování

Kryptografie či šifrování je nauka o utajení informace převedením do podoby nečitelné bez speciální znalosti – klíče. Šifrovací algoritmy v moderní kryptografii všeobecně dělíme podle klíče na symetrické a asymetrické. Prvně jmenované užívají stejný klíč pro šifrování a dešifrování, druhé podle určitého vzorce vytvářejí dvojici veřejného a soukromého klíče. Zvláštní kategorii tvoří proudové šifry, které nějakým způsobem vytvářejí „key stream“ a kódují data po bitech. Tyto šifry se řadí mezi symetrické.

V našem případě se bude jednat o přenášení hovoru zabezpečeného proti odposlechnutí jeho obsahu třetí osobou. Konkrétní šifrovací mechanismy, jejich výhody a nevýhody budou zváženy dále [HAC].

I Nyquistův teorém zvaný též: Shannonův teorém, Nyquistův-Shannonův teorém, Shannonův-Nyquistův-Kotělnikův teorém: „*Přesná rekonstrukce spojitého, frekvenčně omezeného, signálu z jeho vzorků je možná tehdy, pokud byl vzorkován frekvencí alespoň dvakrát vyšší než je maximální frekvence rekonstruovaného signálu.*“

1.4 FITkit

Platforma FITkit je vývojová a demonstrační deska založená na mikrokontroléru z rodiny MSP430 od Texas Instruments (dále MCU) a programovatelném hradlovém poli Spartan 3 od Xilinxu (dále FPGA). Deska je osazena různými periferiemi a rozhraními za účelem maximální variability nasazení.

1.5 Problematika telefonních linek

Telefonní linky jsou přenosovým médiem pro přenos hlasových a datových volání. V začátcích byla přenášená data pouze hlasem, v dnešní době jsou po telefonních linkách přenášena i jiná data. Původní klasická vedení byla čistě analogová, ale vzhledem ke snahám o sdílení prostředků – linek byl zaveden zpočátku frekvenční multiplex (FDM). Pokusně byl určen rozsah frekvencí lidského hlasu, který je rozhodující pro srozumitelnost. Byla definována tzv. „telefonní kvalita“. Tento rozsah frekvencí je 0,3 – 3,4 kHz.

Budeme-li uvažovat vzorkování signálu, tak budeme na základě Nyquistova teorému potřebovat vzorkovací frekvenci přibližně 6,8 kHz. Tato frekvence se ovšem v praxi nepoužívá a běžná je frekvence 8 kHz, která je spojována s telefonní kvalitou komprese PCM.

1.6 Modem

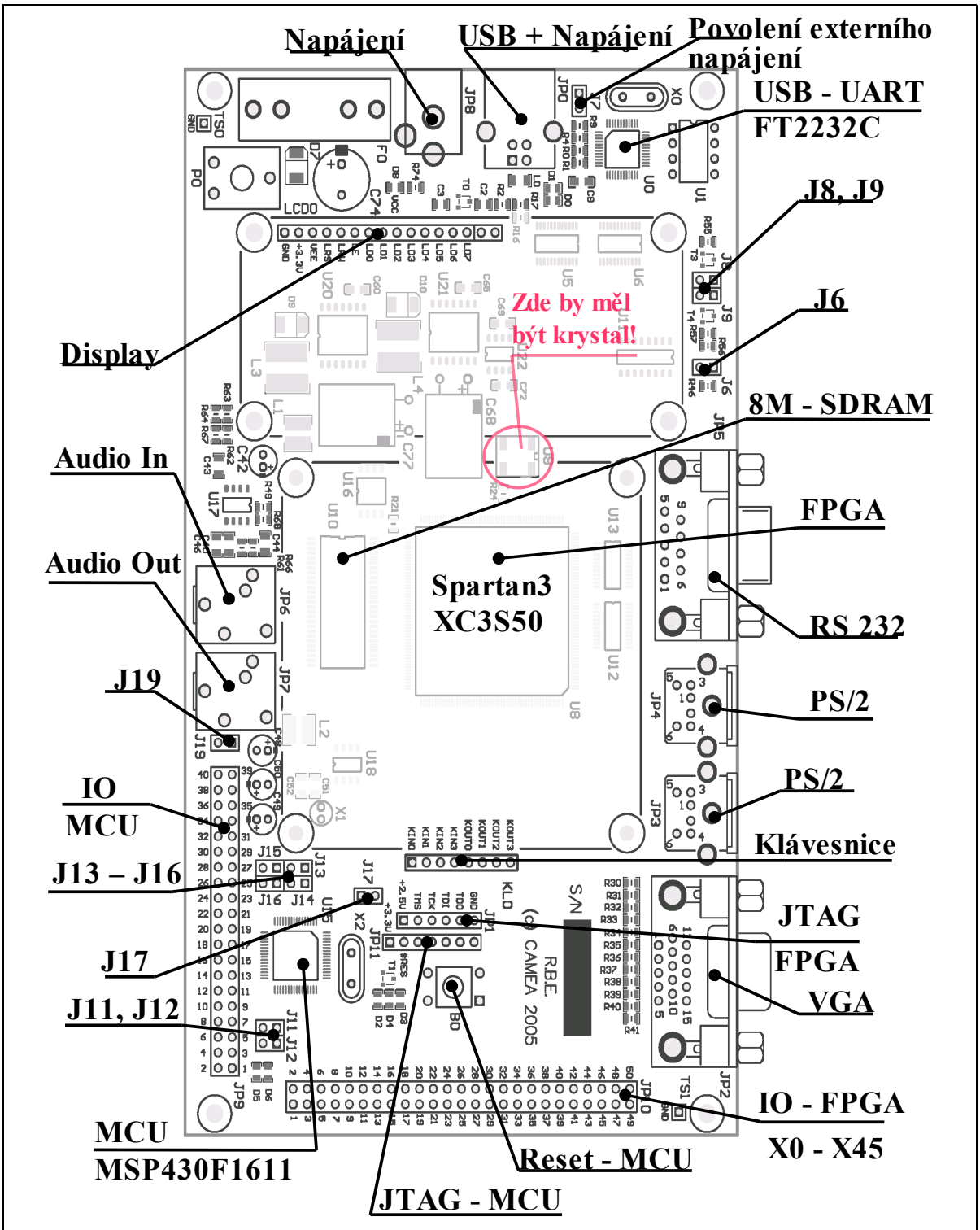
Modem je slovo vzniklé zkrácením a spojením slov modulátor a demodulátor. Všeobecně se jedná o zařízení převádějící signál z digitálního na analogový a obráceně.

Kresba 1: Užitý modem DeskPorte Home

V našem případě se bude jednat o klasický telefonní modem Microcom DeskPorte Home. Tyto modemy se starají o převod dat z digitální podoby do analogové a užívají různá kanálová kódování například kvadraturně amplitudovou modulaci (QAM) s různým počtem hladin. Typická maximální rychlost těchto zařízení je 56 kb/s pro downlink a 48kb/s pro uplink. To je také maximální datový tok, do kterého se musíme vejít s námi komprimovaným a šifrovaným zvukem. Modem podporuje korekci chyb (ITU-T V.42 LAPM, MNP 2 – 4) na kterou budeme plně spoléhat.

2 Platforma FITkit

Před vlastním návrhem mapování funkčních bloků na hardware a software se seznámíme podrobně s platformou FITkitu a jejími relevantními částmi.



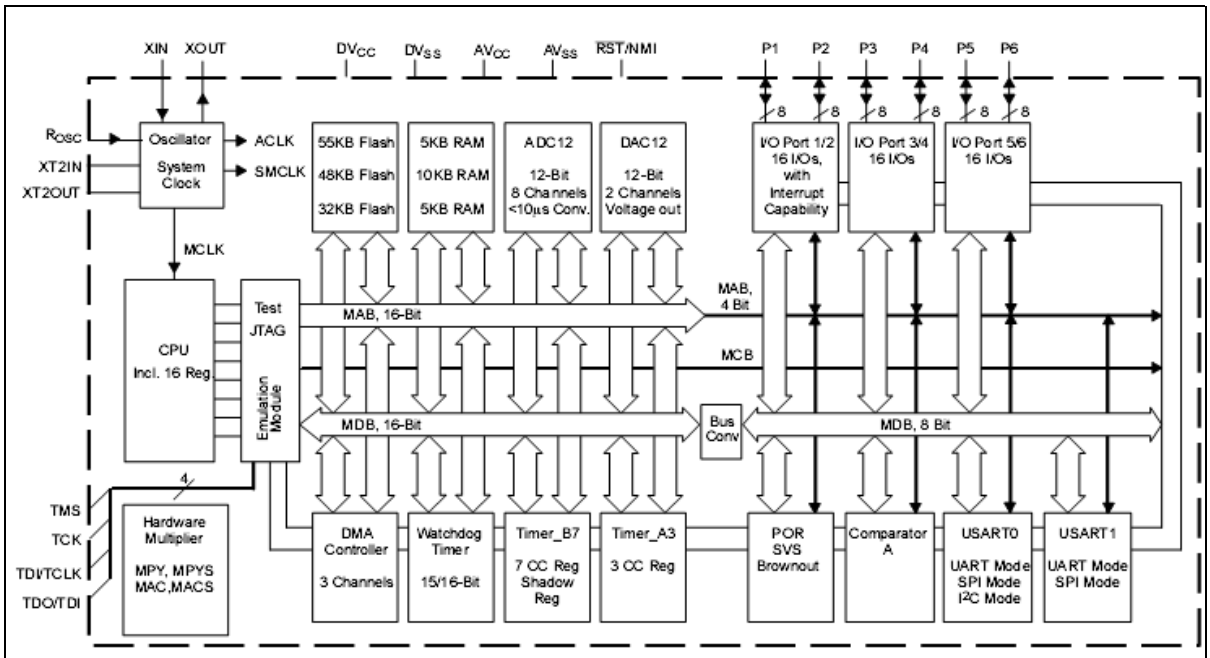
Kresba 2: Deska FITkitu, rozmístění komponent a rozhraní [FKM].

FITkit (dále jen kit) je deska osazená mikrokontrolérem Texas Instruments MSP430F1611, který tvoří výpočetní jádro kitu, na desce je také programovatelné hradlové pole Xilinx Spartan 3 XC3S50 PQ208. Kit se připojuje k PC za účelem programování pomocí USB (Universal Serial Bus) kabelu. K MCU se přistupuje pomocí terminálu otevřeném na čipem FT2232C emulovaném COM portu. Pro audio vstup a výstup je FITkit osazen zásuvkami 3,5 jack.

2.1 Periferie

Na desce je umístěna řada periferií. K nejsnáze viditelným patří textový šestnáctiznakový LCD (Liquid Crystal Display) displej CM1610NR-J2 a maticová klávesnice 4x4. Z dalších částí bych zmínil určitě i čip SDRAM (Synchronous Dynamic Random Access Memory) 8x8Mbit, paměť Flash a několik LED (Light-Emitting Diode) diod [FKM].

2.2 Mikrokontrolér



Kresba 3: Blokové schéma MSP430F1611 [TI1]

Jednotka MCU obsahuje 16bitový mikroprocesor na architektuře RISC (Reduced Instruction Set Computer) s přibližně třiceti instrukcemi. Integrované periferie jsou následující: oscilátor, paměť Flash (48KB + 256B), paměť RAM (10KB), tři šestnáctibitové časovače, z nichž jeden je určen primárně pro modul Watchdog, tříkanálový řadič DMA (Direct Memory Access), dedikovaná násobička pro násobení 16x16, 16x8, 8x16, 8x8 bitů, šest osmibitových vstupně-výstupních portů, komparátor, osmikanálový dvanáctibitový AD převodník s bufferem pro 16 vzorků, dvoukanálový

DA (Digital Analog) převodník, 2 USART (Universal Synchronous Asynchronous Receiver/Transmitter) rozhraní a obvod pro kontrolu napájecího napětí (POR SVS Brownout). V našem případě je jedno USART rozhraní využito jako SPI (Serial Peripheral Interface) pro komunikaci s FPGA a pamětí flash umístěnou na FITkitu. Rozhraní USART může fungovat také v I^2C módu, který využívají například některé monitory [TI1][TI2].

Memory	Size	48KB
Main: Interrupt vector	Flash	0FFFFh - 0FFE0h
Main: Code memory	Flash	0FFFFh - 04000h
RAM (Total)	Size	10KB
		038FFh - 01100h
Extended	Size	8KB
		038FFh - 01900h
Mirrored	Size	2KB
		018FFh - 01100h
Information memory	Size	256 Byte
	Flash	010FFh - 01000h
Boot memory	Size	1KB
	ROM	00FFFh - 00C00h
RAM	Size	2KB
(mirrored at 018FFh-01110h)		009FFh - 00200h
Peripherals	16-bit	001FFh - 00100h
	8-bit	000FFh - 00010h
	8-bit SFR	0000Fh - 00000h

Tabulka 1: Organizace paměti mikrokontroléru msp430f1611

2.2.1 Převodník ADC12

Převodník obsažený v MCU provádí dvanáctibitový převod a je schopen ve své paměti uchovat až 16 vzorků. Umožňuje zároveň vzorkovat až 8 kanálů, v našem případě budou stačit pouze ty dva, které máme připojeny k audiokonektoru [TI1][TI2].

Převodník podporuje vnitřní referenční napětí 1.5V a 2.5V a umožňuje i zapojení vnějšího referenčního napětí.

ADC12 obsahuje vlastní oscilátor, který je možno nastavit až na 6,3 MHz, získání jednoho vzorku trvá minimálně 13 tiků hodin, což nám dává maximální možnou frekvenci vstupního signálu 242 kHz (6,3/13/2 (dle vzorkovacího teorému)). Jak je vidět, tak převod je více než dostatečně rychlý pro vzorkování zvuku (20 Hz- 20 kHz). Oscilátor je možno pro úsporu energie snadno vypínat.

Vzorkování může probíhat ve čtyřech základních módech:

1. Jednorázový převod jednoho kanálu.
2. Jednorázový převod sekvence kanálů.
3. Opakovaný převod jednoho kanálu.
4. Opakovaný převod sekvence kanálů.

Nás mohou zajímat v podstatě všechny tyto módy. Budeme-li se zabývat pouze jedním kanálem, což je nejjednodušší možnost (telefon stejně nemá 2 mikrofony aby podporoval stereo), bude pro nás výhodný mód 1 nebo 3. Osobně se domnívám, že jednodušší bude užití módu jedna, protože bude snadnější napsat obsluhu přerušení. Užití módu 3 nám však vynese rychlejší vzorkování [TI1][TI2].

Vlastní konfigurace ADC12 probíhá pomocí konfiguračních registrů, každé paměťové místo vzorku obsahuje vlastní (ADC12MCTLx), dále jsou zde dva globální konfigurační registry (ADC12CTL0, ADC12CTL1), registr s příznakem přerušení (ADC12IFG), registr povolující přerušení (ADC12IE), registr s vektorem přerušení (ADC12IV) [TI1][TI2].

Z dokumentace [TI1][TI2] je zjevné, že převodník disponuje možností generovat osmnáct různých přerušení. Šestnáct z nich oznamuje konec převodu v jednotlivých pamětech, další je generováno při přepsání nevyzvednuté hodnoty (overflow) a jedno, pokud je vyžadováno vzorkování před ukončením předchozí konverze (timeout).

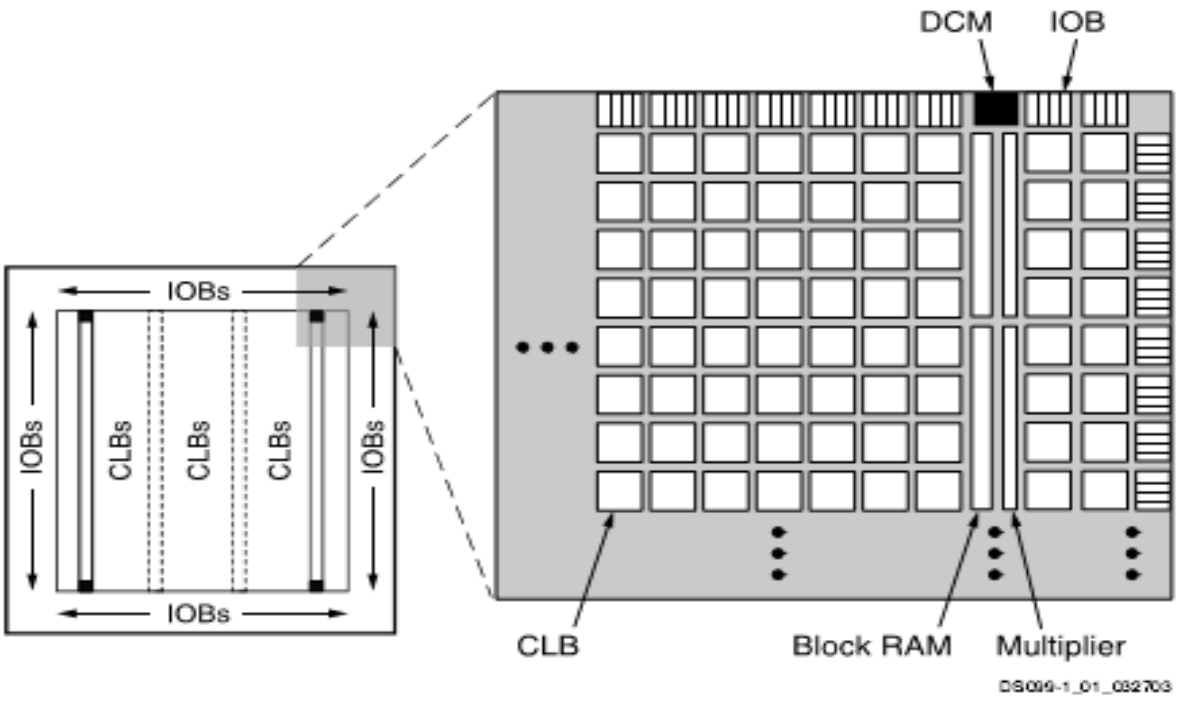
2.2.2 Převodník DAC12

Na čipu MSP430F1611 se nacházejí tyto převodníky dva: jeden pro levý, a jeden pro pravý kanál. DAC12 může pracovat ve 12bitovém i 8bitovém režimu a využívat DMA kanálů. Převodník pracuje s čísly v přímém binárním kódu, nebo ve druhém doplňku a umožňuje kladný i záporný posun výstupního signálu. DAC12 podporuje několik úsporných módů, kdy je převod pomalejší, ale má nižší spotřebu. Tyto módy jsou pro nás irelevantní, jelikož předpokládáme napájení z elektrické sítě.

Při konverzi budeme na oba audio kanály zasílat stejný signál, takže levý i pravý reproduktor (sluchátko) budou přehrávat stejný zvuk.

Ke správné funkci převodníků (AD i DA) v součinnosti se sluchátky a připojeným vstupem je třeba zapojit jumpery J13-J16 do polohy sepnuto. Tímto připojíme audio vstupy a výstupy k pinům na mikrokontroléru. Jumper J19 slouží k zapnutí výstupního audio zesilovače a je třeba ho rovněž uvést do zapnuté polohy.

2.3 FPGA



Kresba 4: Schéma vnitřního uspořádání FPGA [X2]

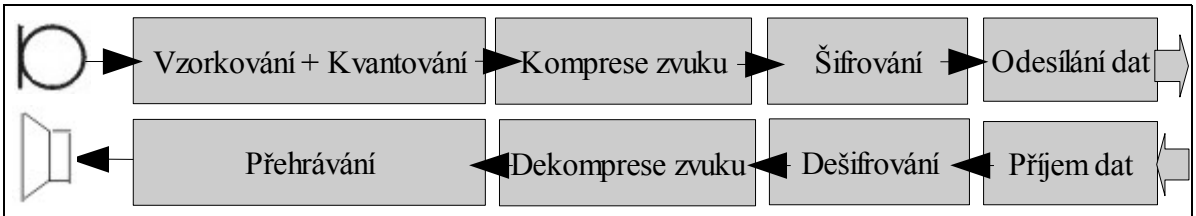
Programovatelné hradlové pole Spartan 3 XC3S50 obsahuje padesát tisíc logických hradel realizovaných vyhledávacími tabulkami na bázi RAM (LUT) a D-klopnými obvody uspořádaných do logických bloků (CLB) v šestnácti řádcích a dvanácti sloupcích. Co se týče distribuované paměti obsahuje FPGA dvanáct kilobitů. Bloky jsou propojeny mezi sebou a s výstupními piny pomocí programovatelných spojů (IOB). Dalšími komponentami tvořícími čip jsou 18Kb bloky RAM (1 sloupec), čtyři násobičky 18x18b a čtyři jednotky správy hodinového signálu (DCM) [FKM][X1][X2][X3].

2.4 Rozhraní

Kit je opatřen následujícími standardními konektory: PS/2 myš, PS/2 klávesnice, RS232 (9 pinů) samec, VGA, Audio IN/OUT (2x 3,5mm stereo jack), dále se zde nachází již dříve zmíněné USB užité k programování a napájecí konektor pro dodatečné napájení. Vstup/výstup je realizován také konektory s 50 (FPGA) a 40 (MCU) samostatnými piny. Po jednom pinu z těchto rozhraní je využito pro přerušení z FPGA. Konkrétně se jedná o pin 26 z JP9 a pin 5 z JP10. Pro programování MCU resp. FPGA jsou na desce umístěny konektory JTAG JP11 resp. JP1. K povolení PC rozhraní je třeba uzavřít jumper J6 [FKM].

3 Analýza problému

Jak již bylo dříve předestřeno, cílem projektu je navrhnout a otestovat systém pro šifrování telefonního hovoru. Vzhledem k tomu, že zvuk je jakožto analogová veličina krajně nevhodný pro šifrování, musíme ho upravit do pro nás vhodnější digitální podoby. K tomuto slouží blok provádějící vzorkování a kvantování. Takto zpracovaný digitální signál je pro nás ovšem stále nevhodný, protože jsme omezeni datovým tokem, který je možné přenášet po telefonní lince. Musíme tedy provést vhodnou kompresi, která sníží datový tok, aniž by došlo k výraznému úbytku srozumitelnosti. Vzhledem k tomu, že všeobecné kompresní algoritmy se pro kompresi zvuku nehodí, využijeme blok s kompresním algoritmem, který je optimalizovaný pro zvuk, nebo lépe přímo pro řeč. Takto zpracovaný signál jsme schopni podrobit vlastnímu šifrování. K šifrování užijeme blokovou šifru, protože předpokládáme, že při chybném přenosu nebudeme data zasílat znovu a proudové šifry jsou náchylnější k desynchronizaci vlivem chyby. Zašifrovaná data pak zbývá přenést po telefonní lince. V současné konfiguraci k tomu slouží modem Microcom DeskPorte Home připojený přes sériové rozhraní.



Kresba 5: Hrubé schéma navrhovaného systému

Na kresbě 5 je zobrazen datový tok v systému. Ovládání související s případným vytáčením čísla, nastavováním klíčů apod. je pro jednoduchost vypuštěno. Systém jsem rozdělil na v kresbě zobrazené funkční bloky. Konkrétní namapování funkčních bloků do hardwaru a softwaru bude dále podrobně rozebráno.

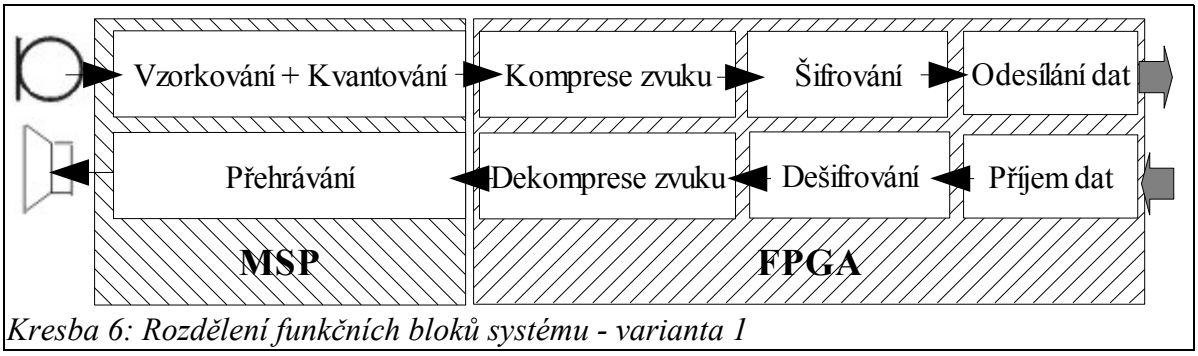
Ve finálním produktu budou ve schématu zakreslený mikrofon – vstup analogového signálu a reproduktor (sluchátko) – výstup analogového signálu nahrazeny linkou připojenou ke klasickému telefonu.

3.1 Mapování systému

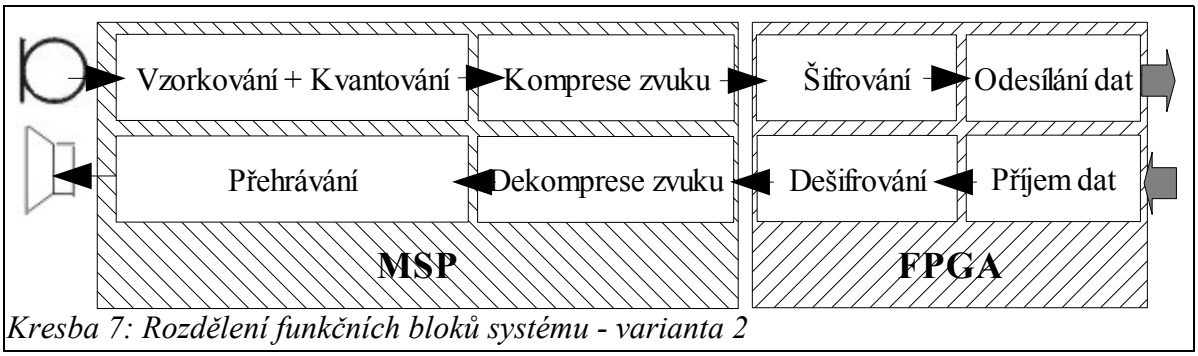
V úvodní analýze problému jsem v kresbě 5 předestřel hrubé schéma sytému, aniž bych rozváděl, která část bude jakým způsobem realizována. Jelikož AD a DA převodníky se nacházejí na

mikrokontroléru, je zcela zjevné, že vzorkování i kvantování budou prováděny právě tam, stejně tak i přehrávání.

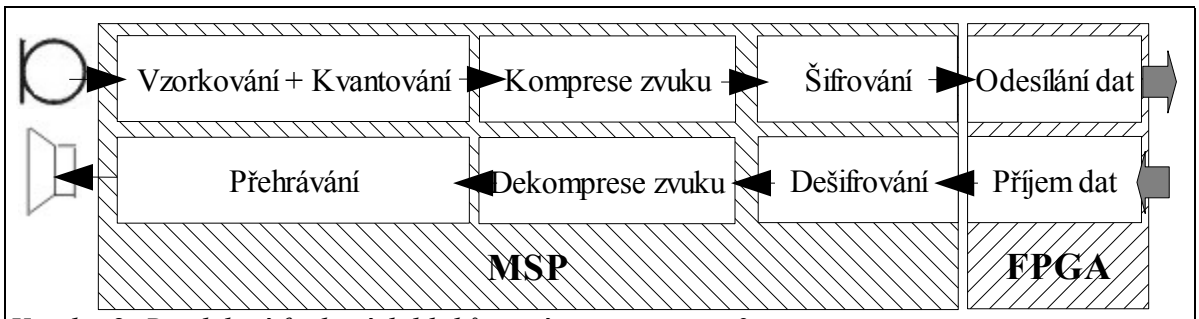
Komunikace po sériové lince s modemem bude naopak probíhat přes FPGA protože konektor rozhraní RS232 v devítipinovém provedení (samec) je připojen právě ke zmíněnému hradlovému poli.



Kresba 6: Rozdělení funkčních bloků systému - varianta 1



Kresba 7: Rozdělení funkčních bloků systému - varianta 2



Kresba 8: Rozdělení funkčních bloků systému - varianta 3

Kompresní a šifrovací bloky můžeme teoreticky umístit do mikrokontroléru jakožto softwarovou implementaci, nebo do FPGA jako návrh hardwaru ve VHDL.

Vzhledem k tomu, že hradlové pole je s mikrokontrolérem propojeno pouze pomocí rozhraní SPI (a pinu pro přerušení) viz. dále, budeme se snažit omezit komunikaci po tomto rozhraní a proto se kompresi pokusíme umístit do mikrokontroléru, abychom dosáhli snížení datového toku.

Blok s šifrováním pak můžeme umístit libovolně, ale preferovaná varianta je rovněž softwarová implementace v mikrokontroléru (kresba 8), aby mohlo být hradlové pole případně ve výsledném produktu vypuštěno.

3.2 Komprese zvuku

V této kapitole zmíním konkrétní východiska zpracování zvuku od analogové podoby až po komprimovaný zvukový signál. Zaměřím se zde na softwarové řešení této problematiky, protože vzhledem k umístění AD převodníku na mikrokontrolér bude minimálně vzorkování a kvantování prováděno právě tam. Mapování komprese na mikrokontrolér je dalším logickým krokem. Protože nepředpokládám řešení komprese pomocí specializovaného čipu, nebudu zde tuto variantu nijak rozvádět.

3.2.1 Vzorkování

Vzorkování bude v souladu se standardem telefonní kvality probíhat na 8 kHz. Z převodníku získáme 12b vzorek. Ten můžeme oříznout na 8bitů, nebo vhodně zkomprimovat. Vzorkovat budeme pouze jeden kanál, protože telefon stejně není stereo. V první fázi budeme komprimovat a posílat všechny vzorky, v pozdějších fázích by bylo vhodné přidat detekci hlasu (VAD), nebo zvuku alespoň v nějaké minimální formě, například pomocí prahu, kdy několik po sobě jdoucích vzorků má podobnou hodnotu blížící se nule. Na straně příjemce by pak v době, kdy nemá přehrávač vzorky, mohl být generován a přehráván šum.

3.2.2 Zvukové kodeky a formáty

Kodek je slovo vniklé spojením kodér a dekodér. Jedná se o dvojici kompresní a dekompresní algoritmus.

Když si odmyslíme kvantizační šum při kvantování signálu, dělíme komprese zvuku na ztrátové a bezztrátové. Bezztrátové komprese jsou pro nás z hlediska datového toku neschůdné, ale přesto budou dále pro úplnost alespoň letmo zmíněny.

Z hlediska způsobu komprese jsou pro nás relevantní dvě skupiny kodeků a sice klasické audio kodeky a kodeky specializované pro řeč – vokodéry (vocoders).

Zatímco klasické zvukové kodeky mají za úkol komprimovat všeobecný zvuk, hlasové kodeky jsou specializované na pásma specifická pro řeč, a často pracují s modelem lidského hlasového ústrojí.

Kodeky založené na diferenciálním principu bude třeba případně modifikovat, protože většinou jsou optimalizované pro kompresi souborů a nikoliv po přenos na tak nespolehlivém médiu, jako je telefonní linka.

V následujících podkapitolách si probereme různé kodeky a formáty z hlediska vhodnosti pro náš systém.

A-law, μ-law

Jsou součástí standardu G.711. Převádí 14, 13bitové vzorky na logaritmické 8bitové. Produkuje 64kb/s datový tok. Hodí nám protože máme 12b AD převodník a ořezem spodních bitů bychom ztratili příliš mnoho informace. Produkovaný datový tok 64kb/s je ale příliš vysoký, takže bude ve výsledku potřeba užít ještě jinou kompresi. Například navazující ADPCM [CS].

AAC

AAC je ztrátový zvukový formát vyvinutý jakožto nástupce formátu MP3. Provádí diskrétní kosinovou transformaci a je výpočetně značně náročný, tudíž se jím nebudeme zabývat [F1].

AC3

AC3 je proprietární ztrátový zvukový formát vyvinutý v Dolby Laboratories. Jelikož usilujeme o produkt nezatížený patenty, nebudeme tento kodek pro naši implementaci uvažovat [ATSC].

DPCM, ADPCM

DPCM pracuje podobně jako PCM, ale posílá pouze rozdíl oproti odhadnuté hodnotě určené na základě předchozích vzorků.

ADPCM jde ještě o krok dál, a přizpůsobuje predikci konkrétnímu hlasu mluvčího. Vzhledem k datovému toku 32kb/s (G.721) 32, 24, 40kb/s (G.726) a 16kb/s (G.727) se tyto kodeky řadí mezi kandidáty pro naší aplikaci [SPC].

FLAC, Monkey Audio

FLAC a Monkey Audio jsou otevřené bezztrátové kompresní formáty. Vysoký a proměnný datový tok je pro nás činí nepoužitelnými [CO][ASH].

GSM & CELP kodeky

Původní algoritmus CELP navržený Schroederem a Atalem vyžadoval pro kompresi 1 sekundy zvuku 100 sekund výpočtu na superpočítači Cray I. Tento fakt jasně ukazuje náročnost CELP algoritmů. Naštěstí už od této doby vývoj pokročil, takže pro nás má smysl se touto rodinou kodeků zabývat.

Náročnost algoritmů je dána tím, že kódování funguje optimalizací v uzavřené smyčce − iterací přes kódová slova a porovnáváním zpětně dekódovaného výstupu se vstupem [VA].

Kodeky na bázi excitované lineární predikce pracují s modelem lidského hlasového a sluchového ústrojí. Snaží se vzniklý šum posunout do frekvencí, na které je lidský sluch méně citlivý.

Do této kategorie patří kodeky GSM, G.728 a Speex. Vhodnost těchto kodeků z hlediska výpočetní náročnosti je třeba posoudit praktickým testováním na FITkitu.

Ogg Vorbis

Jedná se o otevřený ztrátový formát s volně dostupnými implementacemi. Zásadní nevýhodou je pro nás variabilní bitový tok a náročnost výpočtu diskrétní kosinové transformace a inverzní diskrétní kosinové transformace na jejímž principu mimo jiné kodek pracuje. Vorbis navíc využívá hojně plovoucí desetinnou čárku, což je pro výpočet na FITkitu signifikantní nevýhoda [XO].

MP3

MPEG Layer 3 (Motion Picture Expert Group Layer 3) je ztrátový kompresní formát založený na diskrétní kosinové transformaci. Byl vyvinut Fraunhoferovým institutem. Mp3 vypouští pro lidské ucho neslyšitelné, nebo nevýznamné informace. Přílišná náročnost a patenty činí kodek pro naše potřeby nevhodným [BOU].

PCM

PCM je kodek založený na ukládání vzorků nasnímaných konstantní frekvencí na daném počtu bitů. Jedná se o naprosto základní bezztrátový kodek. Pro naše užití je ale nevhodný, protože při 8kHz na 8 bitech produkuje datový tok 64kb/s a navíc máme dvanáctibitový vzorek.

Real Audio

Proprietární zvukový formát využívající různé kodeky jak ztrátové, tak i bezztrátové mimo jiné užívá i kodeky založené na CELP, AAC, AC3 [RA].

Kodek	*Algoritmus*	*Vzorkovací frekvence*	*Datový tok*	*Bitů na vzorek*	*CBR/VBR*	*Stereo/Mono*	*Patenty, poplatky*	*Implementace*
AAC	ztrátový, hybridní	8Hz - 192kHz	8 – 528kb/s	jakýkoliv (fp)	Ano / Ano	Ano / Ano	Ano	Proprietární
ALAC	bezztrátový	44,1 - 48kHz	proměnný	?	Ne / Ano	Ano / Ano	Ne	Proprietární
FLAC	bezztrátový	1Hz - 1048kHz	proměnný	4,8,16,24,32	Ne / Ano	Ano / Ano	Ne	OSI: referenční
Monkey's Audio	bezztrátový	8; 11,025; 12; 16; 22,05; 24; 32; 44,1; 48 kHz	proměnný		Ne / Ano	Ano / Ano	Ne	Proprietární
MP3	ztrátový	8, 11,025, 12, 16, 22,05, 24, 32, 44.1, 48 kHz	8; 16; 24; 32; 40; 48; 56; 64; 80; 96; 112; 128; 192kb/s	jakýkoliv (fp)	Ano / Ano	Ano / Ano	Ano	Proprietární
RealAudio	bezztrátový, ztrátový	různé	proměnný	různý	Ano / Ano	Ano / Ano	Ano	Proprietární
Speex	řeč	8; 16; 32; 48kHz	2,15 - 24,6 kb/s 2,15 - 24,6 kb/s	?	Ano / Ano	Ano / Ano	Ne	OSI: referenční
Vorbis	ztrátový	1 Hz - 200 kHz	proměnný	jakýkoliv(fp)	Ano / Ano	Ano / Ano	Ne	OSI: referenční
WavPack	ztrátový, bezztrátový, hybridní	1 Hz - 16777 kHz	196kBit a více, proměnný	minimálně 2,2	Ano / Ano	Ano / Ano		OSI: referenční
WMA	ztrátový, bezztrátový	8; 11,025; 16; 22,05; 32; kHz	4 to 768kb/s nebo proměnný	16, 24 nebo jakýkoliv (fp)	Ano / Ano	Ano / Ano		Proprietární

Poznámka: některé datové toky a vzorkovací frekvence byly pro přehlednost vynechány.

Tabulka 2: Přehled kodeků a formátů

Shrnutí

V tabulce 2 je vidět, že některé kodeky mají uveden jakýkoliv počet bitů na vzorek, protože používají vnitřně plovoucí desetinnou čárku. Jelikož výpočty v plovoucí desetinné čárce jsou na mikrokontroléru neefektivní, bude vhodné se těmto kodekům vyhnout. Z výběru pro naši aplikaci rovněž vyřadíme patentované a zpoplatněné kodeky.

3.3 Šifrovací algoritmy

Na počátku je třeba se rozhodnout, zda-li budeme užívat symetrické nebo asymetrické šifrovací algoritmy. Vzhledem k výpočetní náročnosti druhých jmenovaných budeme používat symetrickou kryptografii. Protože nepředpokládáme, že by se případný útočník mohl pokoušet o změnu zprávy v průběhu hovoru přeskládáním a modifikací bloků zprávy, použijeme blokovou šifru bez provazování bloků tj. ECB (Electronic Code Book) režim. Dalším důvodem k užití šifry v režimu ECB je snaha zabránit chybě v jenom bloku projevit se vícekrát. [HAC].

Jednou z možných variant šifrování je šifra AES (Rijnadel) v blokovém režimu. Při užití 128b klíče budeme snadno šifrovat úseky dat po 16 osmibitových vzorcích, posléze větší počet vzorků v 128bitových blocích [NIST].

Druhou možnou alternativou je šifra Skipjack. Šifra užívá 80bitový klíč a 64bitové bloky dat, což by mohlo snížit dobu výpočtu oproti AES a jelikož šifrování bude probíhat častěji kratší dobu (sníží se latence), mohla by tato šifrovací metoda být pro naši aplikaci vhodnější. Bohužel se zatím vyzkoušená implementace v praxi neosvědčila [NIST].

Další možností je šifra Blowfish, která pracuje rovněž s 64 bitovými bloky a klíčem o délce až 448 bitů. Šifra se řadí mezi nejrychlejší a velmi pomalé předzpracování klíče nám nijak nevadí. Problém je ovšem s množstvím spotřebované paměti které je oproti jiným šifrám poměrně výrazné [SCHN].

Jedním z dalších zvažovaných kryptografických algoritmů je šifra TEA (Tiny Encryption Algorithm). Je to jeden z nejjednodušších algoritmů a je prostorově i časově nenáročný. Hlavním problémem této šifry je její zaměření na 32bitovou aritmetiku, která je pro nás vzhledem k přítomnosti pouze šestnácti a osmibitové aritmetiky na MCU nedostupná.

Mnohem zajímavější se mi jevila šifra FEAL (Fast Encipherment Algorithm), která je založená na 8bitové aritmetice, protože je ale méně bezpečná než DES, tak jsem ji z výběru taktéž vyřadil [HAC].

Ze zcela jiných důvodů jsem zavrhl šifru IDEA (International Data Encryption Algorithm). Šifru pokrývají patenty pro komerční využití a jelikož výsledný produkt by neměl asi být zdarma, tak jsem v rámci projektu od dalšího bádání nad touto šifrou upustil [HAC].

3.4 Modem

V první fázi testování nám poslouží externí počítačový modem pro sériový port Microcom DeskPorte Home, který byl již dříve zmíněn. Ve finálním produktu předpokládám modemový modul pro vestavěné systémy. Modem budeme ovládat pomocí klasických AT příkazů po sériovém portu. Relevantní pro nás budou zvláště příkazy týkající se vytáčení, příjmu volání a samozřejmě přechodu mezi datovým a konfiguračním režimem.

3.5 Ovládání aplikace

Aplikace bude v první fázi ovládána pomocí počítače, posléze využijeme periferie zabudované na FITkitu. Ve finálním produktu předpokládám, že vytáčení čísla obstará klasický telefon. Zadání klíče pak bude možné provést z připojeného PC, nebo přímo pomocí klávesnice na finálním zařízení. Další detaily budou zmíněny v následujících kapitolách.

3.6 Komunikační protokol

Nedílnou součástí návrhu podobných zařízení je definice protokolu, jímž komunikující strany ustavují spojení. Existuje velké množství různých protokolů, kde dochází k jednostranné i oboustranné autentizaci.

Zajímavým by mohl pro nás být Diffie-Hellmanův algoritmus ustavení klíče. Tento protokol umožňuje bez předchozí společné znalosti ustavit nový klíč pro symetrickou kryptografii a to i v nezabezpečeném kanálu [HAC].

Základním problémem u vestavěných zařízení jako je FITkit, který se projevuje v asymetrické kryptografii i v protokolu Diffie-Hellman, je nedostupnost kvalitního generátoru náhodných čísel. Generátor není dostupný jako funkce v kompilátoru pro mikrokontrolér a jeho implementace v softwarové podobě by byla pseudonáhodná a to ještě velmi málo protože jsme omezeni na šestnáctibitová čísla. Výše zmíněný protokol navíc vyžaduje generování prvočísel, což je vzhledem k dostupnému výpočetnímu výkonu nemožné a uložení tabulky prvočísel by vyžadovalo paměť, které máme na MCU už tak málo.

Jak je vidět, tak implementace komunikačního protokolu je příliš náročná a z ní plynoucí výhody jsou v porovnání s nevýhodami zanedbatelné a mizivé.

Komunikační protokol nebude tedy na úrovni zařízení, ale na úrovni uživatele, který se dohodne se svým protějškem po nešifrovaném kanále, jestli a od kdy budou šifrovat předem dohodnutým klíčem.

4 Implementace

V této kapitole a jejích podkapitolách shrneme současný stav vývoje a testované varianty řešení. Zaměříme se zde také na zhodnocení problémů, které se během implementace vyskytly.

4.1 Mapování systému

V současné implementaci je rozdělení systému realizováno jako na kresbě 8 na stránce 13. Hlavní zátěž tedy nese mikrokontrolér a na FPGA jsou pouze řadiče periferií.

4.2 Vzorkování a komprese

V naší aplikaci probíhá vzorkování jednoho kanálu. Dvanáctibitový vzorek je komprimován pomocí a-law na 8bitů.

Jednu z implementací kodeku GSM (1992 Jutta Degener and Carsten Bormann z Berlínské univerzity - http://kbs.cs.tu-berlin.de/~jutta/toast.html) jsem otestoval na FITkitu, ale ukázala se příliš náročnou a zvuk byl příliš zkreslený, až nesrozumitelný i bez zátěže šifrování.

Kodek LD CELP G.728 – implementace Alexe Zatsmana editovaná Michaelem Concannonem (http://www-mobile.ecs.soton.ac.uk/speech_codecs/standards/g728.html), se při praktickém testování rovněž ukázala příliš náročnou i bez následného šifrování.

Kodeky na bázi ADPCM se ukázaly při praktickém testu jako příliš náročné a neúměrně zpomalily celý proces.

Speex, otevřený kodek na bázi CELP vyvíjený v Xiph.org. jsem doposud neotestoval, ale patří mezi potenciální kandidáty na kompresní algoritmus systému. Domnívám se ovšem, že se v případě testování rovněž ukáže jako příliš náročný.

4.3 Šifrování

V průběhu testování jsem vyzkoušel implementaci algoritmu Skipjack optimalizovanou Markem Tillotsonem pro procesory RISC. Tato varianta se ovšem po mnohahodinovém ladění ukázala nepoužitelnou. Problém byl v neotestované alokaci paměti pro subklíč. Autor se neobtěžoval ověřit, zda-li alokace proběhla úspěšně. Poté co jsem doplnil testování alokace jsem zjistil, že požadovanou paměť (10x256B) prostě nedostanu. Pokus alokovat paměť staticky se ukázal rovněž marným. Podobně neúspěšně skončily i některé jiné pokusy o připojení kompresních či kryptografických knihoven, množství staticky alokovaných proměnných bylo v součtu příliš velké.

V současné chvíli je implementována šifra AES se 128b klíčem. Šifrování probíhá po blocích, vždy když se nashromáždí blok 128bitů (16 vzorků). Dešifrování a dekódování probíhá obdobně po blocích. Výsledné zpoždění je ale téměř neúnosné.

Pro každý směr komunikace může být jiný klíč. Ve výsledku předpokládám, že bude pouze samostatný klíč pro každou dvojici komunikujících zařízení.

Klíč se bude do zařízení nahrávat z počítače po emulovaném sériovém portu, nebo bude možné ho zadat pomocí klávesnice 4x4 na FITkitu.

Případné úložiště klíčů by mohlo být umístěno na paměti flash, která se nachází na FITkitu. Jelikož flash není nijak zabezpečena proti útočníkovi, který by se fyzicky mohl k zařízení dostat, bylo by vhodné klíče ještě zašifrovat klíčem, který by musel uživatel zadávat při každém použití zařízení. Otázkou je, zda-li by se takovéto opatření vyplatilo a zda-li by nebylo pro uživatele snadnější zadávat prostě klíč pro každé volání a při vypnutí zařízení ho zapomenout. Výsledné zařízení by také kromě testovacích prototypů nemělo podporovat funkci „RAM D“ (viz. dále), která umožňuje výpis paměti RAM MCU obsahující zadaný klíč.

4.4 Spojovací systém FITkitu – SPI

Jak již bylo předestřeno dříve, tak mikrokontrolér a programovatelné hradlové pole jsou propojeny rozhraním SPI. Na tomto rozhraní se také nachází flash paměť. Kvalita implementace tohoto rozhraní je pro nás důležitá, protože od ní se odvíjí zátěž na MSP a hlavně rychlost, s jakou budeme schopni data zasílat modemu.

V dokumentaci k implementaci firmwaru od Ing. Zdeňka Vašíčka je uvedeno, že rozhraní teoreticky umožňuje přenosovou rychlost až 460.8 kB/s. I když budeme uvažovat reálnou rychlost mnohem menší, stále je rozhraní dostatečně rychlé pro naši aplikaci [FKT1].

4.5 Periferie na FPGA

Periferie na FPGA byly implementovány členy FITkit týmu a zdrojové kódy byly modifikovány jen minimálně. Kódy spadají pod připojenou licenci.

4.5.1 Klávesnice 4x4

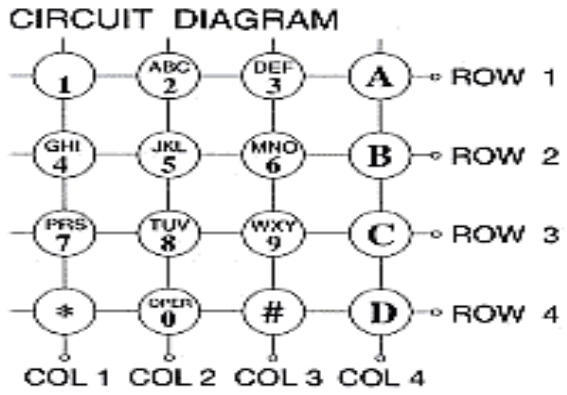
Jak již bylo dříve zmíněno, na kitu je umístěna klávesnice 4x4 typu AK-1604-A-WWB. Klávesy jsou uspořádány do matice. Díky tomuto uspořádání je možné snadno číst stisknuté klávesy buďto po sloupcích, nebo pro řádcích. Čtení v našem případě probíhá následovně: Na vodič čteného řádku je přivedena logická nula, na ostatní řádkové vodiče logická jednička. Při stisknutém tlačítku se

případná logická nula přenese na výstup, nestisknutá klávesa je reprezentována vysokou impedancí a je pomocí PULLUP rezistoru konvertována na logickou jedničku [FKT2].

Řadič klávesnice se ve firmwaru vyskytuje ve verzi s přerušením i bez něj. V našem případě přerušení využijeme.

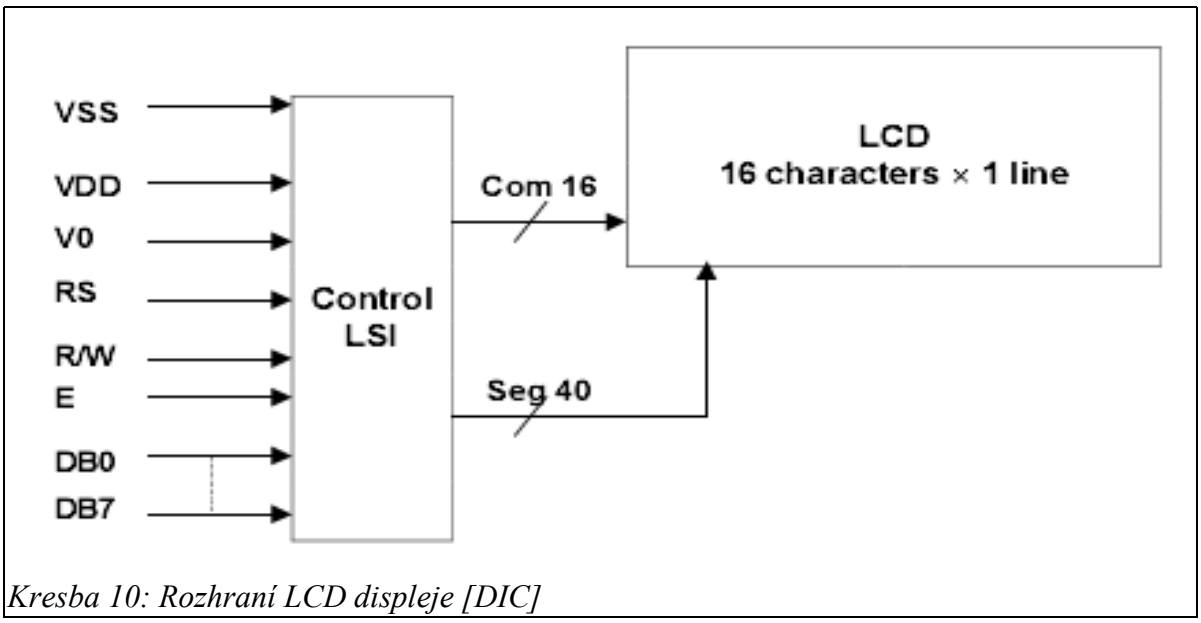
Číslo pinu	*Význam pinu*	*Pojmenování pinu*
1	COL 1	kout0
2	COL 2	kout1
3	COL 3	kout2
4	COL 4	kout3
5	ROW 1	kin0
6	ROW 2	kin1
7	ROW 3	kin2
8	ROW 4	kin3

Tabulka 3: Přiřazení pinů klávesnice



Kresba 9: Diagram klávesnice 4x4 [MCD]

4.5.2 LCD Display



Kresba 10: Rozhraní LCD displeje [DIC]

Kit osazen jednořádkovým šestnáctiznakovým LCD displejem CM1610NR-J2. Displej je řízen příkazy (Příloha A: Příkazy LCD displeje) a je ovládán komponentou, která neumožňuje čtení z displeje. Díky tomuto nemůžeme zjistit, co displej zobrazuje, jak je definován uložený font, ani získat informaci o zaneprázdněnosti displeje. Většina těchto informací ovšem není nezbytná k běžnému používání displeje. Dalším omezením je funkčnost komponenty pouze pro signál

s frekvencí nižší než 8MHz. Toto rovněž nikterak neomezuje užití displeje pro naše účely. Komponenta Display control je samozřejmě připojena přes svůj vlastní adaptér ke sběrnici SPI [DIC].

Displej v naší aplikaci využijeme k zobrazení volaného telefonního čísla, stavu hovoru i k případnému zobrazení zadávaného šifrovacího klíče. V současné chvíli displej zobrazuje „P“ nebo „T“ v závislosti na tom, zda-li je zvolena pulsní, nebo tónová volba.

4.5.3 Řadič RS232

Řadič RS232 je pro naši aplikaci klíčový z hlediska komunikace s modemem. Současná implementace řadiče se ukázala s modemem ne zcela kompatibilní. Dochází občas k chybám v přenosu. Navíc je určena spíše pro sériový port FPGA, který je zapojen jako druhý port na USB.

Jelikož port přes USB se chová jako DCE (Data Circuit-terminating Equipment) a vyvedený port se chová jako DTE (Data Terminal Equipment), jsou zde rozdíly ve směru signálů RTS (Request To Send) a CTS (Clear To Send). Ve výsledku ale na směru signálů RTS a CTS nezáleží, protože ač jsou do řadiče zapojeny, není s nimi nijak zacházeno. Z toho vyplývá, že v řadiči není řízení toku pomocí těchto signálů nijak ošetřeno stejně jako není ošetřeno řízení pomocí XON a XOFF.

Řadič RS232 je implementován s přerušením při přijatém i odeslaném znaku. Řadič není vybaven bufferem, takže je třeba posílat znaky na něj po jednom, což by se mohlo ukázat, jako problém a zbytečné zpomalení pro mikrokontrolér.

Obsluha „přerušení z FPGA“ i terminálu je řešena v hlavní smyčce a může docházet k jejímu vyhladovění. Přerušení z FPGA obsluhované z hlavní smyčky v současné době způsobuje značně nejistou a nespolehlivou obsluhu uživatelských akcí jako stisk klávesy na klávesnici 4x4 nebo i zápis příkazu na terminálu z PC. Tyto rutiny bude třeba ošetřit jinak, nebo nějak zajistit programu spolehlivý návrat do hlavní smyčky, který obslouží události.

4.6 Ovládání aplikace

4.6.1 Klávesnice 4x4

Klávesnici 4x4 jsou jsou přiřazeny následující funkce:

0-9 – zadání telefonního čísla.

A – přepínání pulsní a tónové volby (indikováno na displeji)

B – smazání displeje a zadaného čísla.

C –vytočení čísla

D – zavěšení.

- iniciační příkazy pro modem.

* - nepřiřazeno.

Konkrétní chování by se mohlo ještě upravit a rozšířit v závislosti na stavu hovoru. Například před zahájením hovoru by určité klávesy umožňovaly zadat klíč. Při běžícím hovoru by mohla klávesa „C“ způsobit zavěšení. Iniciační příkazy by se mohly provádět autonomně v závislosti stav modemu a podobně.

4.6.2 Příkazy terminálu

Následující uživatelské příkazy jsou k dispozici na terminálu:

SEND <string>	Zápis řetězce <string> na modem.
*<string>	Zápis řetězce <string> na modem (bez výpisu).
CALL <number>	Vytočení čísla <number> na modemu.
INIT	Iniciace nastavení modemu.
PULSE	Nastaví pulsní volbu.
TONE	Nastaví tónovou volbu.
RUN	Spustí AD/DA převod.
STOP	Zastaví AD/DA převod.
RSREAD	Načtení jednoho znaku z rs232 na FPGA.
RSWRITE <c>	Zápis znaku <c> na rs232 na FPGA.
DISCLEAR	Smazání displeje.
DISCHAR	Zápis znaku na displej.
DIS string	Zápis řetězce na displej.
WIRE	Přepíná šifrovací režim.

0 – Nic

1 – Vzorek jde ze vstupu rovnou na výstup.

2 – Vzorek je komprimován a dekomprimován.

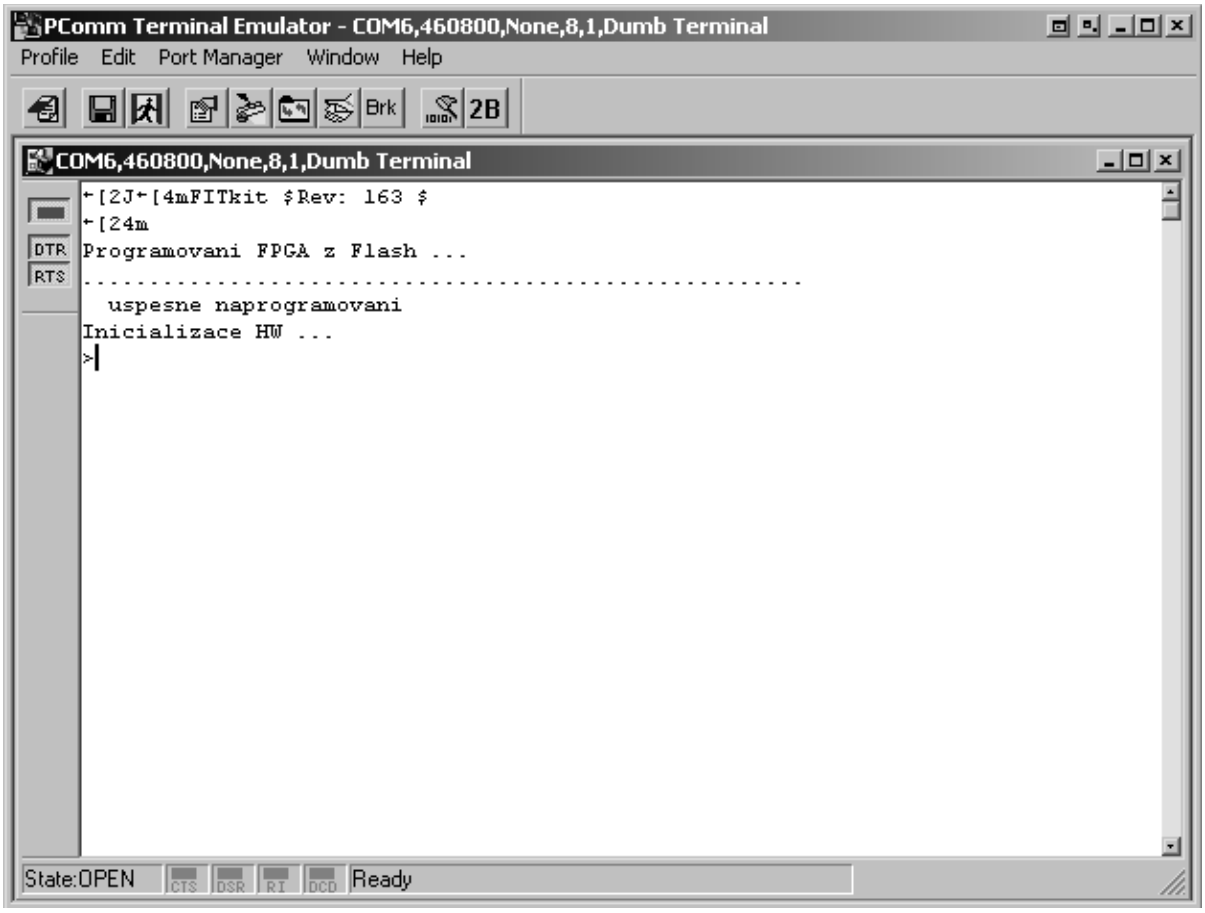
3 – Vzorek je po kompresi „šifrován“ operací XOR s klíčem.

4 – Vzorek je po kompresi šifrován kryptografickým algoritmem.

Následující systémové příkazy jsou k dispozici na terminálu:

PROG FPGA.	Programováni FPGA z PC (XModem).
PROG FPGA FLASH	Programování FPGA z paměti flash.
RESET MCU	Softwarový reset MCU.
RESET FPGA	Softwarový reset FPGA.
CLS	Reset terminálu.
FLASH W C <c> adr	Provede zápis znaku <c> do flash na adresu 'adr'.

FLASH W X adr	Provede zápis dat do flash na adresu 'adr' (XModem).
FLASH W FPGA	Zápis FPGA dat z PC do flash (XModem).
FLASH R A adr	Výpis 64Bytu flash z adresy 'adr'.
FLASH R P page	Výpis stránky c. 'page' z flash.
FLASH R B block	Výpis bloku c. 'block' z flash.
FLASH R S	Vypíše status registru flash.
FLASH E FPGA	Odstraní FPGA data z flash.
FLASH E A adr	Smaže byte ve flash na adrese 'adr'.
FLASH E P page	Smaže stránku 'page' z flash.
FLASH E B block	Smazání bloku 'block' z flash.
FLASH E ALL	Smazání celé flash.
RAM D	Výpis obsahu RAM na terminál.



Kresba 11: Terminál FITkitu po startu.

Jak je vidět, tak škála příkazů pro terminál je velmi bohatá a obsahuje příkazy pro práci snad se všemi periferiemi na FPGA.

4.7 Implementační problémy

V této kapitole nastíním problémy a omezení, na něž jsem narazil v rámci implementace. Zhodnotím zde závažnost nalezených problémů a případné možné řešení.

4.7.1 Velikost paměti

Zcela zásadním omezením se v rámci implementace ukázala velikost paměti flash a RAM na mikrokontroléru (podrobnosti Tabulka 1 strana 9). Při pokusech užít různé kryptografické knihovny nebo knihovny komprese zvuku jsem narazil na obdobné chybové hlášení kompilátoru:

```
msp430-ld: address 0xd52 of output.elf section .bss is not within region data
msp430-ld: address 0xd52 of output.elf section .noinit is not within region data
```

Jak jsem vypátral na diskusních fórech na internetu je tento problém způsoben nedostatkem paměti na mikrokontroléru pro staticky alokované proměnné a linkované knihovny.

U některých knihoven jsem nedostal ani tuto chybovou hlášku a mikrokontrolér během činnosti prostě zatuhl. Knihovny s tímto problémem jsem odložil „k ledu“ než naleznu příčinu, což se mi podařilo u implementace Skipjacku od Marka Tillotsona. Společným jmenovatelem problému byla dynamická alokace paměti příkazem malloc, u níž se neověřovala úspěšnost, a následný pokus o přístup do nealokované paměti.

Jenom pro porovnání: Jedna z nejmenších aplikací (testled), užívající pouze základní knihovny, má po zkompilování 43924B. Vzhledem k velikosti paměti programu, která činí přibližně 49152B (tabulka 1), to není zanedbatelná velikost a přidáme-li ještě knihovny ovládání periferií, nezbývá prostě na šifrovací a kompresní knihovny už příliš místa. Řešení tohoto problému vidím jedině snad ve změně platformy a využití výkonnějšího mikrokontroléru, nebo přesunu funkcí na FPGA. Rád bych zdůraznil, že FITkit sice obsahuje paměťový čip SDRAM, ale ten je připojen přes FPGA a jeho využití by tudíž bylo mnohonásobně pomalejší, čímž se pro nás stává irelevantním.

4.7.2 Ladicí výpisy

Zásadní nevýhodou při ladění na FITkitu oproti PC se ukázala nemožnost odesílat větší množství ladicích výpisů rychle za sebou. Při pokusech získat rozsáhlejší ladicí výpis odeslaný na terminál PC docházelo často zatuhnutí FITkitu. Toto si vysvětluji jako nedostatečnou kapacitu front znaků případně, což je méně pravděpodobné, jako překročení rychlosti rozhraní terminálu (460800Baudů).

5 Testování systému

Tato kapitola se zbývá metodami a technikami, jimiž byl nebo by mohl být v budoucnu systém testován.

5.1 Testy komprese zvuku

První částí bylo otestování kompilovatelnosti kompresní knihovny překladačem pro mikrokontrolér a následné přilinkování k projektu.

Do vstupu kitu jsem připojil výstup hudby a mluveného slova z PC, a do výstupu jsem zapojil sluchátka. Poté co jsem otestoval funkčnost vzorkování výstupem z ADC12 posílaným přímo na DAC12, bylo třeba ověřit funkčnost komprese zvuku. To jsem provedl kompresí získaného vzorku, okamžitou dekompresí a zasláním vzorku na DAC12.

5.2 Testy šifrování

Než proberu testovací zapojení, které jsem užíval v součinnosti s kompresí abych otestoval celkovou funkčnost kompresně-šifrovacího systému, zmíním ještě metodiku testů, které jsem prováděl před větší integrací šifrovacího algoritmu do systému.

V první řadě jsem zkusil, zda-li se mi podaří knihovnu vůbec zkompilovat a přilinkovat. Pokud jsem slavil úspěch, vytvořil jsem testovací vektor, který jsem se pokusil algoritmem zašifrovat a následně dešifrovat. Následně jsem vypsal na terminál původní, šifrovaný a dešifrovaný řetězec. Takto jsem posuzoval funkčnost šifrovacího algoritmu. Náročnost jsem testoval až posléze pomocí testovacího zapojení.

5.3 Testovací zapojení

5.3.1 První fáze – testování komprese a šifrování

Pro potřeby testování jsem audio vstup FITkitu připojil na výstup počítače a přehrával na něm hudbu a mluvené slovo, výstup jsem zapojil do sluchátek. Výstup z kódování a šifrování jsem předal rovnou do dešifrování a přehrávání. Tímto způsobem jsem byl schopen testovat správnou funkčnost komprese zvuku i šifrování a případná přílišná náročnost testované implementace se projevila naprostou nesrozumitelností výsledného zvuku. Docházelo k přerušení obslužné rutiny přerušení obsluhou následujícího vzorku a vyhladověním hlavní smyčky a obsluh uživatelských událostí.

5.3.2 Druhá fáze – testování komunikace po telefonní lince

Tato fáze neproběhla, pouze bylo otestováno vytočení čísla FITkitem a prozvonění telefonu připojeného k ústředně. Vytočení proběhlo pomocí klávesnice na kitu.

V druhé fázi, pokud k ní někdy dojde, bude třeba testovat systém po stránce komunikace po telefonní lince pomocí modemu. Za tímto účelem by mohla být využita telefonní ústředna v laboratoři. Až v této fázi bude zjevné, nakolik se implementace zdařila a zda-li je kvalita zvuku dostatečná pro srozumitelnou komunikaci.

5.4 Kvalita zvuku

Kvalitu zvuku jsem v posuzoval na základě subjektivního poslechu mluveného slova a hudebních skladeb.

5.5 Úzká hrdla

Za hlavní úzké hrdlo systému považuji vlastní telefonní linku, která má za ideálních podmínek 56kb/s zpravidla však méně. Právě tato rychlost je určující pro návrh celého systému. Původně jsem měl obavu i z rozhraní SPI, ale naštěstí se nepotvrdila.

6 Hardwarová implementace

Tato kapitola si klade za cíl specifikovat požadavky a změny potřebné pro hardwarovou implementaci.

6.1 Změny v hardwaru

Vzhledem k doporučení, že ve výsledku by neměla pokud možno být FPGA bude potřeba buďto produkt osadit aplikačně specifickým čipem (ASIC) jehož návrh bude vyplývat z konfigurace FPGA, nebo doplnit desku o jeden či více mikrokontrolérů ovládajících periferie. Toto vyplývá z nízkého počtu vývodů mikrokontroléru a jeho nedostatečného výkonu. Vzhledem k ceně výroby ASIC se mi jeví schůdnější cesta mikrokontrolérů ovládajících periferie.

Výsledné zařízení bude muset být osazeno modemovým modulem pro vestavěné systémy a dodatečnou elektronikou, která bude přepínat mezi přímým propojením linky a telefonu a propojením šifrovanou cestou. Je pro nás značně nevýhodné provádět rozpoznávání vytáčeného čísla a pak toto vytáčení nechávat na modemovém modulu. Mnohem elegantnějším se mi jeví nechat uživatele navázat spojení a až poté přepnout na šifrovaný přenos.

Vzhledem k odlišným napěťovým úrovním na telefonní lince nebude možné užít AD a DA převodníky mikrokontroléru a bude třeba kromě modemového modulu osadit i externí převodník nebo přímo nějaký modul s rozhraním na telefonní linku.

FITkit je napájen z portu USB nebo pomocí externího adaptéru s napětím 5V. Naše zařízení může být napájeno externím adaptérem a bude pravděpodobně vyžadovat doplňující elektroniku pro různá napětí potřebná pro různé moduly (Modem, AD/DA etc.). Alternativou může být napájení z telefonní linky ale osobně bych tuto alternativu nevolil, protože ji považuji za příliš komplikovanou.

6.2 Změny v softwaru

Z předchozí kapitoly je zjevné, že změny v softwaru budou značné. Už v prvním odstavci je zmíněno, že FPGA bude nahrazeno nejspíše mikrokontroléry. Tato změna se projeví v softwaru kompletní přestavbou komunikace s periferiemi, které bychom se vyhnuli užitím ASIC. Můžeme tedy zapomenout na většinu softwarových knihoven, které periferie ovládají a bude třeba dopsat software na mikrokontrolér(y) spravující periferie a modifikovat software na MCU pro komunikaci s těmito koprocesory.

Vzhledem k integraci modemového modulu bude třeba přepsat komunikaci přímo na konkrétní modul, v rámci kitu probíhala tato skrze FPGA a řadič RS232, což ve výsledném produktu zcela jistě nebude.

Další velkou změnou bude využití externích AD, DA převodníků. Vzhledem značné provázanosti softwaru s přerušeními integrovaných převodníků bude potřeba celý program od základu přestavět. Alternativou by bylo přerušení z externího převodníku mapovat na přerušení mikrokontroléru a upravit obslužné rutiny na toto.

7 Závěr

V současné chvíli jsou na FITkitu implementovány všechny potřebné řadiče periferií. Případné vylepšení a odladění by bylo potřeba provést snad jen u řadiče RS232.

Vzorkování funguje správně, komprese A-law je rovněž funkční, i když bude třeba ji nahradit nějakou metodou s vyšším kompresním poměrem. Otestovaná komprese G.726, která patřila mezi kandidáty se ukázala nepoužitelnou. Kodek GSM se jeví příliš náročným alespoň co se týče testované implementace. CELP komprese, kterou jsem vyzkoušel, se rovněž ukázala býti nevhodnou. Použitelný algoritmus, který by nebyl příliš velký a náročný zůstává nadále otevřenou otázkou.

Šifrování pomocí AES v režimu ECB je funkční, ale je příliš náročné a bude třeba ho buď ještě nějak zoptimalizovat, nahradit nějakou lepší implementací, nebo jinou vhodnou šifrou. Veškeré pokusy o jeho optimalizaci (jako převod operací xor na 16b verze v zájmu snížení počtu instrukcí a zrychlení operací) se ukázaly nedostatečné. Otestoval jsem šifru Skipjack, bohužel, ač se na první pohled jevila mnohem výhodnější, získaná implementace byla paměťově příliš náročná a obsahovala programátorské chyby.

Komunikace s modemem, vytáčení a příjem hovorů je funkční. Po úplném odladění komprese zvuku a šifrování mělo přijít na řadu vlastní testování celého systému pomocí propojení dvou FITkitů skrze modemy a telefonní ústřednu. V současné chvíli se mi jeví zhola nemožné, uskutečnit úspěšně variantu v původně navržené podobě – tedy s šifrováním i kompresí na mikrokontroléru. Jelikož jsem opakovaně narážel na nedostatek paměti pro program a to i u jednodušších algoritmů, domnívám se, že bude třeba od této varianty upustit.

V rámci řešení komprese i šifrování jsem nalezl množství slepých uliček a odhalil množství komplikací jež bude třeba ke zdárnému řešení projektu překonat. Zdá se, že projet v původně zamýšlené podobě, jak je na kresbě 8 ze strany 13 je nerealizovatelný. Větší šance na úspěch by byla v případe umístění šifrovacích rutin na FPGA jak je to na kresbě 7. Šifrování by díky obvodovému paralelismu bylo jistě mnohonásobně rychlejší. Další možností je poohlédnout se po koprocesoru, který by se ujal komprese a šifrování s tím, že současné MCU by se staralo jen o řízení.

Budeme-li se zabývat využitelností současné implementace pro budoucí produkt, dospějeme k závěru, že po softwarové stránce bude třeba závažných změn, které nelze provést bez dostupného prototypu zařízení. K navržení tohoto prototypu bude třeba široké mezioborové spolupráce mezi informatiky, elektrotechniky a konečně i designérem, který navrhne atraktivní design nového produktu v závislosti na funkčních požadavcích.

Zůstává otázkou, jak dlouho, budou se současným trendem přechodu na VoIP technologie a sítě třetí generace, klasické telefony ještě významným segmentem trhu.

Literatura

[AMZ] HROMČÍK, Petr. *Analýza možnosti zabezpečení telefonního spojení*. [semestrální projekt], Brno, FIT VUT v Brně, 2002

[ASH] ASHLAND, Matthew T. *Monkey's Audio - a fast and powerful lossless audio compressor*. [online] změněno 8.1.2006 v 11:22 [citováno 30. 3. 2008]
dostupný z WWW: <http://www.monkeysaudio.com/theory.html>

[ATSC] ATSC: *Digital Audio Compression Standard (AC-3, E-AC-3)Revision B*[online].Advanced Television Systems Committee, Inc.,1750 K Street, N.W., Suite 1200, Washington, D.C. 20006, změněno 7.2005 v 17:34 [citováno 30. 3. 2008] dostupný z WWW:
<http://www.atsc.org/standards/a_52b.pdf>

[BOU] BOUVIGNE, Gabriel. *MP3'Tech - www.mp3-tech.org* [online]. změněno 21.1.2007 v 19:19 [citováno 30. 3. 2008] dostupný z WWW:<http://www.mp3-tech.org/>

[BPO] VÁVRA, Jakub. *Osciloskop* [bakalářská práce], Brno, FIT VUT v Brně, 2006

[CO] COALSON, Josh. *FLAC - Free Lossless Audio Codec* [online]. změněno 18.3.2008 v 00:12 [citováno 30. 3. 2008] dostupný z WWW:<http://flac.sourceforge.net/>

[CS] CISCO Systems: *Waveform Coding Techniques* [online]. změněno 2.2.2006 [citováno 30. 3. 2008] dostupný z WWW:
<http://www.cisco.com/warp/public/788/signalling/waveform_coding.html>

[DIC] DATA IMAGE CORPORATION. *LCD Module Specification CM1610NR-J2* [online]. změněno 4.6.2000 [citováno 26.12.2007]
dostupný z WWW: <http://www.dataimage.com.tw/product/cm/pdf/CM1610NR-J2.PDF>

[F1] FRAUNHOFER IIS: *Low Delay Audio Coding* [online]. [citováno 30. 3. 2008] dostupný z WWW:<http://www.iis.fraunhofer.de/EN/bf/amm/projects/lowdelay/index.jsp>

[FKM] FUČÍK, Otto, Dr. Ing.. *FITkit Pokyny pro práci* [počítačový soubor: FITkit_guide_20060127.pdf]. FIT VUT Brno, změněno 27.1.2006 [citováno 23.12.2007] <odkaz již není k dispozici>

[FKT1] VAŠÍČEK, Zdeněk. *Firmware / Propojovací systém FITkitu* [online], [citováno 26.12.2007] dostupný z WWW: <http://merlin.fit.vutbr.cz/FITkit/docs/firmware/spifitkit.html>

[FKT2] MARKOVIČ, Ján. *Firmware / Klávesnice 4x4* [online], [citováno 26.12.2007] dostupný z WWW: <http://merlin.fit.vutbr.cz/FITkit/docs/firmware/20060226key.html>

[HAC] MENEZES, Van Oorschot, Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997.

[MCD] 4MOST. *Standard Matrix Circuit Diagram* [online]. změněno 3.4.2006 [citováno 26.12.2007] dostupný z WWW: <http://www.4most.co.uk/accord_standard_matrix.htm>

[NIST] NIST.gov - Computer Security Division - Computer Security Resource Center. *Block Ciphers* [Online] změněno 22.10.2007 [citováno 15.4.2008] dostupný z WWW: <http://csrc.nist.gov/groups/ST/toolkit/block_ciphers.html>

[RA] REAL Networks. *Codecs > RealAudio 10* [online], [citováno 30.3.2008] dostupný z WWW: <http://www.realnetworks.com/products/codecs/realaudio.html>

[SCHN] SCHNEIER, Bruce. *The Blowfish Encryption Algorithm* [online], [citováno 28.12.2007] dostupný z WWW: <http://www.schneier.com/blowfish.html>

[SPC] WOODARD, Jason. *Speech Coding* [online]Department of Electronics & Computer Science, University of Southampton, změněno 9. září 1998 12:29:51 [citováno 28.12.2007] dostupný z WWW: <http://www-mobile.ecs.soton.ac.uk/speech_codecs/index.html>

[TI1] TEXAS INSTRUMENTS. *SLAS368D* [online]. změněno 5.2005 [citováno 23.12.2007] dostupný z WWW: <http://focus.ti.com/lit/ds/symlink/msp430f1611.pdf>

[TI2] TEXAS INSTRUMENTS. *MSP430x1xx User's Guide* [online]. změněno 28.2.2006 [citováno 23.12.2007] dostupný z WWW: <http://focus.ti.com/lit/ug/slau049e/slau049e.pdf>

[VA] VALIN, Jean-Marc. *Speex: A free codec for free speech* [online počítačový soubor], CSIRO ICT Centre, Cmr Vimiera & Pembroke Roads, Marsfield NSW 2122, Australia, Xiph.Org Foundation [citováno 30.3.2008] dostupný z WWW: <http://people.xiph.org/~jm/papers/speex_lca2006.pdf>

[WP1] WIKIPEDIE: Otevřená encyklopedie. *Zvuk* [online]. změněno 20.12.2007 v 18:44 [citováno 22. 12. 2007] dostupný z WWW: <http://cs.wikipedia.org/wiki/Zvuk>

[X1] XILINX. *Spartan-3 FPGA Family Complete Data Sheet* [online]. změněno 2005 [citováno 23.12.2007]. dostupný z WWW: <http://www.xilinx.com/bvdocs/publications/ds099.pdf>

[X2] XILINX. *Spartan-3 Overview* [online].změněno 24.3.2006 [citováno 23.12.2007].dostupný z WWW:
<http://www.xilinx.com/products/silicon_solutions/fpgas/spartan_series/spartan3_fpgas/overview.htm>

[X3] XILINX. *Spartan-3 User Guide Equivalent Documentation* [online]. změněno 25.3.2006 [citováno 23.12.2007]. dostupný z WWW:Maks o
<http://www.xilinx.com/bvdocs/userguides/ug_spartan3.htm>

[XO] XIPH.Org: *Vorbis I specification* [online] změněno 20.7.2004 v 07:19 [citováno 30. 3. 2008] dostupný z WWW: <http://www.xiph.org/vorbis/doc/Vorbis_I_spec.html>

Rejstřík zkratek

AAC	-	Advanced Audio Coding
ADC	-	Analog Digital Converter (v textu také jako AD převodník)
ADPCM	-	Adaptive Differential Pulse Code Modulation
AES	-	Advanced Encryption Standard
ASIC	-	Application Specific Integrated Circuit
CTS	-	Clear To Send
CELP	-	Code Excited Linear Prediction
CLB	-	Configurable Logic Block
DCE	-	Data Circuit-terminating Equipment
DCM	-	Digital Clock Manager
DPCM	-	Differential Pulse Code Modulation
DRAM	-	Dynamic Random Access Memory
DTE	-	Data Terminal Equipment
FLAC	-	Free Lossless Audio Codec
FPGA	-	Field Programmable Gate Array
IDEA	-	International Data Encryption Algorithm
IOB	-	Input Output Block
JTAG	-	Joint Test Action Group
LCD	-	Liquid Crystal Display
LUT	-	Look Up Table
MCU	-	Micro Controller Unit
MP3	-	MPEG-1 Audio Layer 3
PCM	-	Pulse Code Modulation
RISC	-	Reduced Instruction Set Computer
RLE	-	Run Length Encoding
RTS	-	Request To Send
SDRAM	-	Synchronous Dynamic Random Access Memory
SPI	-	Serial Peripheral Interface
USB	-	Universal Serial Bus
VGA	-	Video Graphics Array
VHDL	-	VHSIC Hardware Description Language
VHSIC	-	Very High Speed Integrated Circuit

Přílohy

Příloha A: Příkazy LCD displeje

Instrukce	***Kód instrukce***										***Popis***	***Trvání*** *fosc= 270KHz*
	RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0		
Smazání displeje	0	0	0	0	0	0	0	0	0	1	Zapíše „20h“ do DDRAM a nastaví DDRAM adresu na „00h“.	1,53mS
Kurzor na začátek	0	0	0	0	0	0	0	0	1	-	Nastaví adresu DDRAM na „00h“ a posune kurzor na počáteční pozici. Obsah DDRAM není měněn.	1,53mS
Nastavení režimu	0	0	0	0	0	0	0	1	I/D	SH	Přiřadí směr pohybu kurzoru a nastaví posun celého displeje.	39µS
Nastavení displeje zap./vyp.	0	0	0	0	0	0	1	D	C	B	Nastaví kontrolní bity (D) displej, (C) kurzor, (B) blikání kurzoru zapnuto/vypnuto.	39µS
Změna posunu displeje / kurzoru	0	0	0	0	0	1	S/C	R/L	-	-	Nastaví bit posunu kurzoru, nastaví posun displeje. Nemění obsah DRAM.	39µS
Nastavení funkce	0	0	0	0	1	DL	N	F	-	-	Nastavení: šířky dat DL:8bitů/4bity, počtu řádků N:2/1, fonty F:5x11/5x8.	39µS
Nastav adresu CGRAM	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Nastaví adresu CGRAM v adresovém čítači.	39µS
Nastav adresu DDRAM	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Nastaví adresu DDRAM v adresovém čítači.	39µS
Čti Busy flag a adresu	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Při čtení flagu BF je možné číst i adresu.	0µS*
Zapiš data do RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Zápis do RAM (CGRAM/DDRAM)	43µS
Čti data z RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Čtení z RAM (CGRAM/DDRAM)	43µS
* Po tomto příkazu je třeba vyčkat ½ fosc poté co padne příznak BF do „0“.												

Tabulka 4: Přehled příkazů LCD [DIC]

Příloha B: BSD Licence

Tato licence se vztahuje na materiály a zdrojové kódy FITkit týmu.

LICENSE TERMS

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software or firmware must display the following acknowledgement:

   This product includes software developed by the University of Technology, Faculty of Information Technology, Brno and its contributors.
4. Neither the name of the Company nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

This software is provided "as is", and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the company or contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

Příloha C: Šifrovací zařízení

Enigma 100

http://www.hightech-store.com/enigma.html

TSD 3600

http://www.dutchguard.com/ATT-TSD-3600-telephone-security-device-p-persec.html
http://www.flickr.com/photos/21746901@N08/2275723713/

ELE-900

http://www.electromax.com/ele900.html

SV-100 Safe Voice Scrambler

http://www.pimall.com/nais/telephonescrambler.html
http://www.spyshops.ca/securevoicetelephonescrambler.htm

Phone Line Scrambler Call Encrypter Kit TCS-E1

http://www.brickhousesecurity.com/phonescrambler-voiceencryption.html

KV1200 Landline Scrambler

http://www.spycatcheronline.co.uk/kv1200-landline-scrambler-p-224.html?spyid=ba12743317617bcb0cf5aa06636f924b

Příloha D: Struktura adresářů

Adresář v němž se nachází snímek svn s použitými verzemi knihoven budeme při dalším popisu považovat za kmen a cesty budou relativní vůči němu. Zmíním adresáře víceméně jen přehledově a nebudu rozepisovat všechny konkrétní adresáře, ale zaměřím se na ty, které jsou z nějakého důvodu pro nás zajímavé.

Struktura adresářů je následující:

apps – adresář s uživatelskými aplikacemi, kromě naší aplikace obsahuje i několik dalších..

Phone – naše aplikace.

sw – adresář s programem pro MCU.

top – adresář s konfigurací FPGA.

base – adresář se součástí všeobecných makefilů.

copyright – licenční smlouvy ke knihovnám FITkitu.

ctrls – knihovny periferií na FPGA ve VHDL.

doc – dokumentace (spíše teoreticky).

chips – architektura a omezení FPGA.

models – simulační modely pro periferie na FPGA.

sw – zdrojové kódy knihoven pro FITkit

libfitkit – knihovny pro MCU v jazyce C.

libs – knihovny pro práci s periferiemi na FPGA z MCU.

tests – testovací programy.

units – knihovny ve VHDL pro FPGA.

Příloha E: Nahrávání programu

Nástroje

K úspěšnému zkompilování kódu pro FPGA a MCU na kitu je třeba několik programů a nástrojů:

Ise Webpack – aplikační balík umožňující kompilovat zdrojové kódy ve VHDL do podoby binárního konfiguračního řetězce pro FPGA.

Popis instalace: http://merlin.fit.vutbr.cz/FITkit/docs/navody/20060201ise.html.

MSPgcc – balík kompilátoru a nástroje pro nahrání programu do mikrokontroléru.

Popis instalace: http://merlin.fit.vutbr.cz/FITkit/docs/navody/20060201c.html.

Ovladače pro FTDI Chip – ovladače pro konverzi USB na dva sériové porty.

Popis instalace: http://merlin.fit.vutbr.cz/FITkit/docs/navody/20060201b.html.

Je třeba nastavit čísla portů, aby odpovídala nastavení v souboru ./base/Settings.inc nebo soubor upravit.

Kompilace konfiguračního souboru pro FPGA

Z příkazové řádky přejdeme do adresáře /apps/Phone/top a spustíme příkazem make generování konfigurační informace pro FPGA. Průběh bude vypisován do konzole. Výstupem by měl být soubor output.bin. Více na http://merlin.fit.vutbr.cz/FITkit/docs/navody/makefile.html

Kompilace programu pro MCU

Z příkazové řádky přejdeme do adresáře /apps/Phone/sw a spustíme příkazem „make load“ kompilaci a nahrávání programu do MCU. Pokud je vše správně nastavené a žádný program není připojen na port FITkitu, bude nahrání potvrzeno. je potřeba mít při nahrávání uzavřené jumpery J8 a J9.

Nahrání konfigurace do FPGA

Připojíme se pomocí aplikace hyperterminál nebo PComm Terminal Emulator na port MCU (nižší z portů FITkitu). Nastavíme rychlost na 460800 Baudů, 8 bitů, jeden stopbit a žádnou paritu. Fitkit by se měl přihlásit. Do terminálu zadáme „FLASH W FPGA“ a vybereme zaslání souboru přes xterminal. Zvolíme soubor outpit.bin. Po úspěšném nahrání souboru restartujeme FITkit pomocí vypnutí signálů DTR a RTS v terminálu.